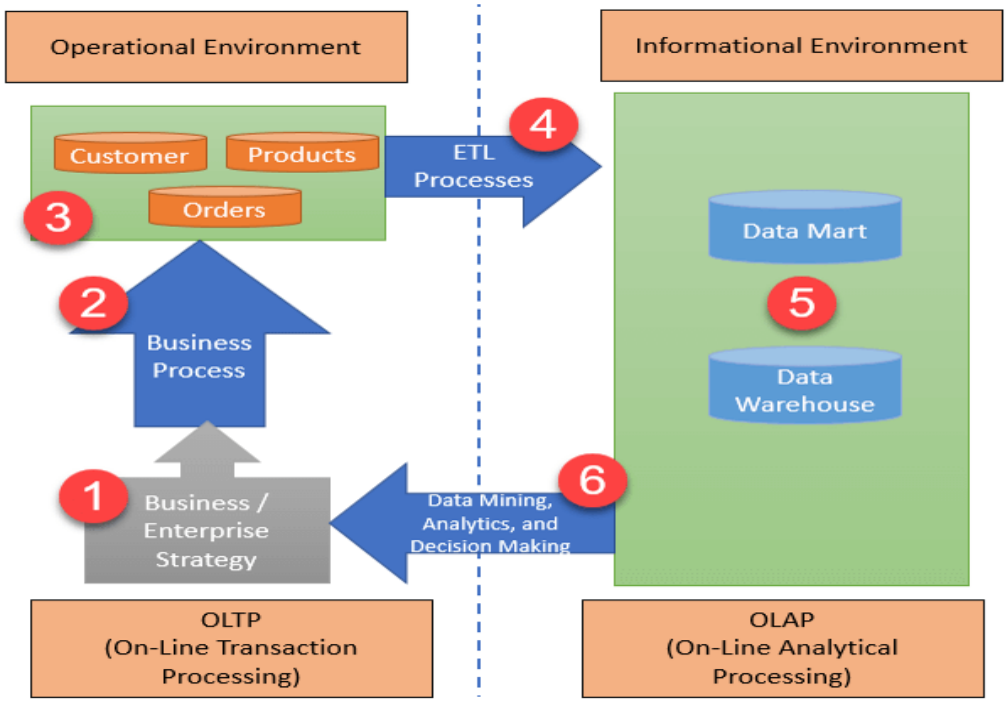
Date⇒ 22-02-2021
Module⇒ Backend
Lecture By⇒ Akash Handa
Subject ⇒Postgres

IN PREVIOUS LECTURE **(QUICK RECAP)** **Date-19/02/2021**	**In Today's Lecture (Overview)**
Aggregate in Mongodb	What is Oltp? Postgres Commands Select PostgreSQL SELECT statement syntax Primary Key Insert into WHERE

What is Oltp?

OLTP is defined as an operational system that supports transaction-oriented applications in a 3-tier architecture. OLTP uses transactions that include small amounts of data. OLTP system is an online database changing system.



Postgres Commands

Select

One of the most common tasks, when you work with the database, is to query data from tables by using the `SELECT` statement.

The `SELECT` statement is one of the most complex statements in PostgreSQL. It has many clauses that you can use to form a flexible query.

Because of its complexity, we will break it down into many shorter and easy-to-understand tutorials so that you can learn about each clause faster.

PostgreSQL SELECT statement syntax

Let's start with the basic form of the `SELECT` statement that retrieves data from a single table.

The following illustrates the syntax of the `SELECT` statement:

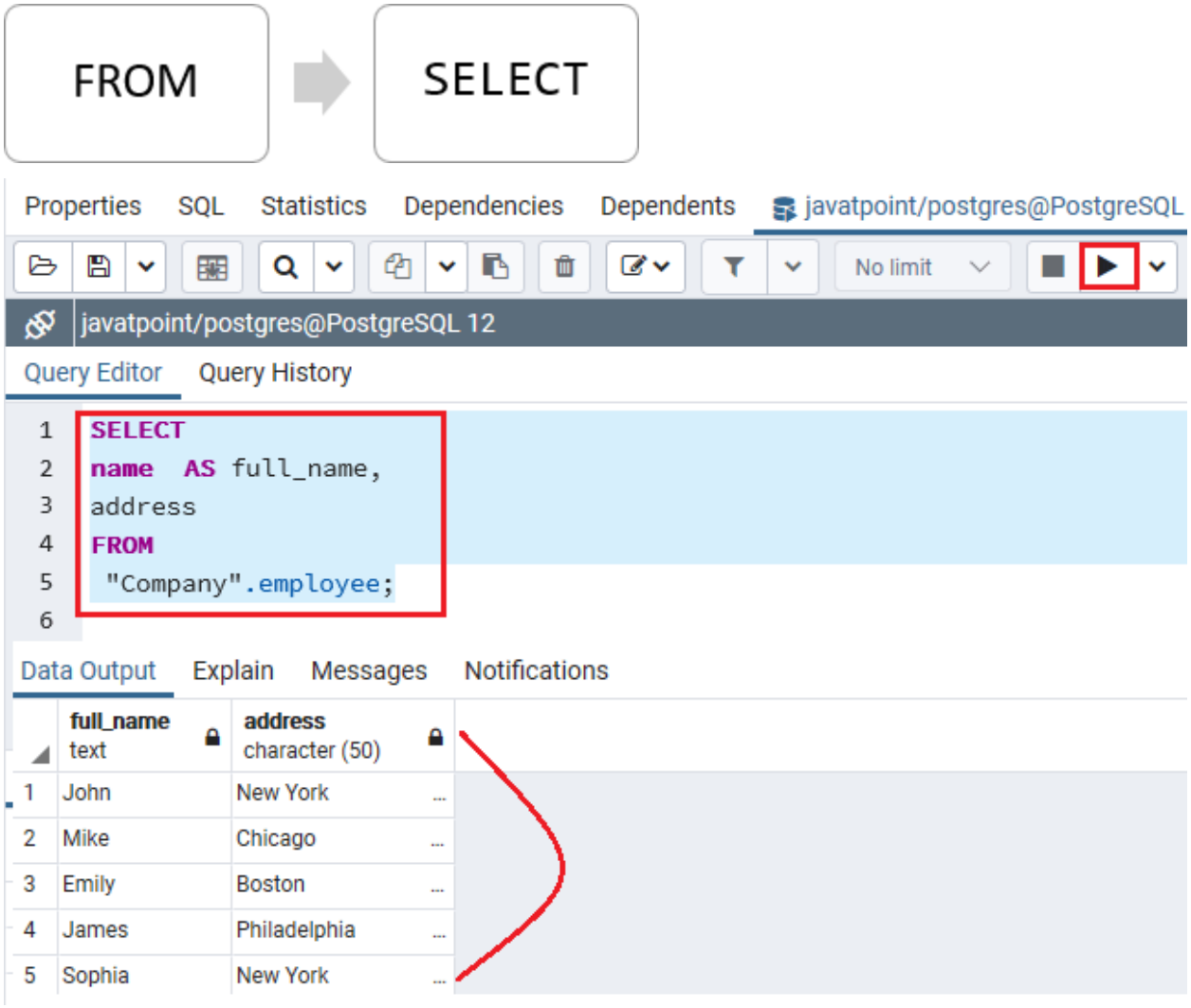
```
SELECT
   select_list
FROM
   Table_name
```

Let's examine the `SELECT`statement in more detail:

> First, specify a select list that can be a column or a list of columns in a table from which you want to retrieve data. If you specify a list of columns, you need to place a comma (`,`) between two columns to separate them. If you want to select data from all the columns of the table, you can use an asterisk (`*`) shorthand instead of specifying all the column names. The select list may also contain expressions or literal values.
> Second, specify the name of the table from which you want to query data after the `FROM` keyword.

The `FROM` clause is optional. If you do not query data from any table, you can omit the `FROM` clause in the `SELECT` statement.

PostgreSQL evaluates the `FROM` clause before the `SELECT` clause in the `SELECT` statement:



Primary Key

primary key is a column or a group of columns used to identify a row uniquely in a table.

You define primary keys through primary key constraints. Technically, a primary key constraint is the combination of a not-null constraint and a UNIQUE constraint.

A table can have one and only one primary key. It is a good practice to add a primary key to every table. When you add a primary key to a table, PostgreSQL creates a unique B-tree index on the column or a group of columns used to define the primary key.

Define primary key when creating the table

Normally, we add the primary key to a table when we define the table's structure using CREATE TABLE statement.

```
CREATE TABLE TABLE (
      column_1 data_type PRIMARY KEY,
      column_2 data_type,
      …
);
```

The following statement creates a purchase order (PO) header table with the name `po_headers`.

```
CREATE TABLE po_headers (

      po_no INTEGER PRIMARY KEY,

      vendor_no INTEGER,

      description TEXT,

      shipping_address TEXT

);
```

Insert into

The PostgreSQL `INSERT` statement allows you to insert a new row into a table.

The following illustrates the most basic syntax of the `INSERT` statement:

```
INSERT INTO table_name(column1, column2, …)
VALUES (value1, value2, …);
```

In this syntax:

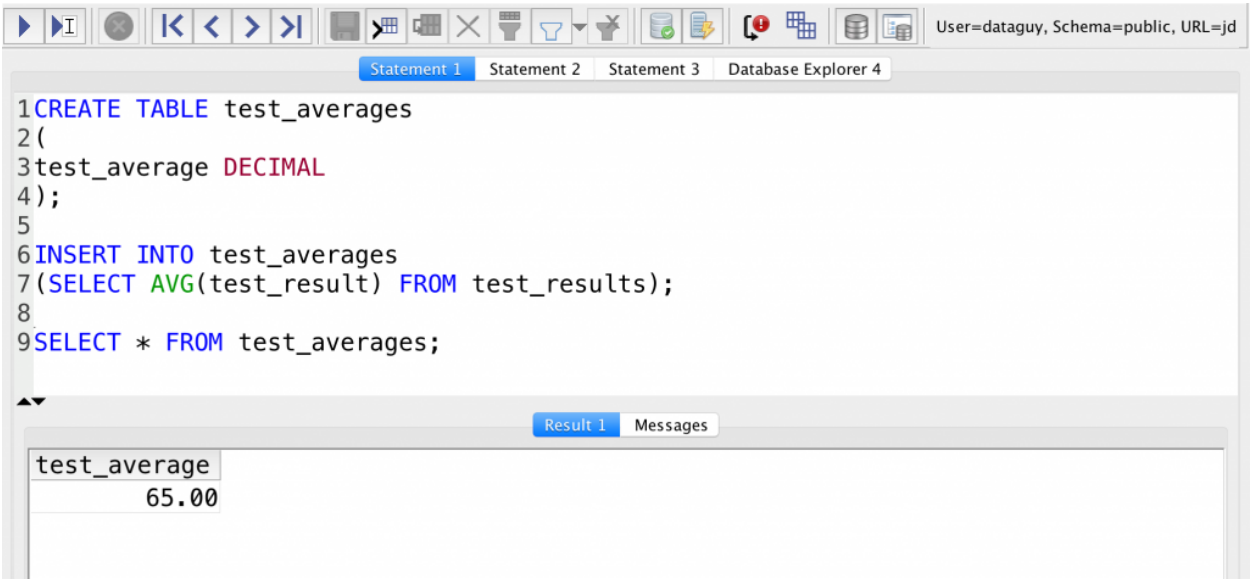
> First, specify the name of the table (`table_name`) that you want to insert data after the `INSERT INTO` keywords and a list of comma-separated columns (`colum1, column2, ....`).

Second, supply a list of comma-separated values in a parentheses `(value1, value2, ...)` after the `VALUES` keyword. The columns and values in the column and value lists must be in the same order.

The `INSERT` statement returns a command tag with the following form:

```
INSERT oid count
```

OID is an object identifier. PostgreSQL used the OID internally as a primary key for its system tables. Typically, the INSERT statement returns OID with value 0. The count is the number of rows that the INSERT statement inserted successfully.



WHERE

The syntax of the PostgreSQL `WHERE` clause is as follows:

```
SELECT select_list
FROM table_name
WHERE condition
ORDER BY sort_expression
```

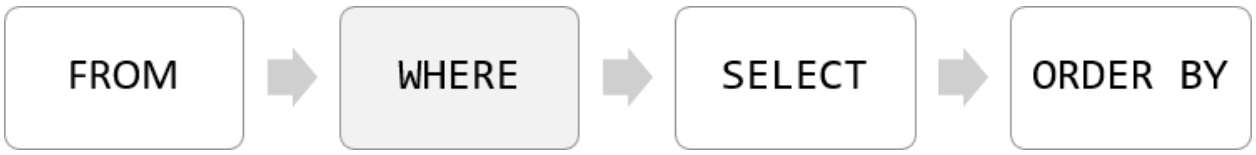
```
Code language: SQL (Structured Query Language) (sql)
```

The `WHERE` clause appears right after the `FROM` clause of the `SELECT` statement. The `WHERE` clause uses the `condition` to filter the rows returned from the `SELECT` clause.

The `condition` must evaluate to true, false, or unknown. It can be a boolean expression or a combination of boolean expressions using the `AND` and `OR` operators.

The query returns only rows that satisfy the `condition` in the `WHERE` clause. In other words, only rows that cause the `condition` evaluates to true will be included in the result set.

PostgreSQL evaluates the `WHERE` clause after the `FROM` clause and before the `SELECT` and `ORDER BY` clause:




```
SELECT
	last_name,
	first_name
FROM
	customer
WHERE
	first_name = 'Jamie';
```

	last_name character varying (45)	first_name character varying (45)
1	Rice	Jamie
2	Waugh	Jamie